SPECIFICATION

Electronic Version 1.2.8
Stylesheet Version 1.0

REVERSE AND FORWARD MULTIPLE HOPPING USING A PROXY WHERE AT LEAST ONE HOP IS SECURE

Background of Invention

[0001] This invention relates generally to establishing a secure connection between a client and a server, and more particularly to establishing a multiple-hop secure connection between the client and the server through an intervening proxy or gateway. At least one of the hops between the client and the proxy and the proxy and the server is secure.

[0002] Secure communication between a client and a server over the Internet has become increasingly important. For example, in the context of electronic commerce, the user at the client is likely only willing to send credit card and other sensitive information to the server through a secure connection. One of the more popular ways to establish a secure connection between a client and a server is through what is known as a secure socket layer (SSL) session. Like other ways to establish secure connections, SSL sessions rely on security keys, or certificates. Generally, these ways to establish secure connections are referred to as part of a particular public key infrastructure (PKI), or part of a symmetric key infrastructure.

[0003] For example, a client wanting to establish a secure connection with a server via an SSL session initially negotiates a session key with the server. The session key enables the client to encrypt data so that it can be securely communicated to the server over the Internet, or other unsecure network. The encrypted data is received at the server, which decrypts the data by using the session key. Thus, what is

known as an SSL tunnel is established between the client and the server, so that the client and the server can securely exchange information. The actual secure exchange of information using SSL over the Internet is usually performed through what is known as the secure hypertext transport protocol (HTTPS).

[0004] Paradoxically, the secure exchange of information can pose a security risk in other ways. For example, the server with which the client is communicating may be part of a private network, such as an intranet. Data from the Internet may be received at a proxy or firewall of the private network, which then distributes the data to the intended server within the network. The distribution is accomplished because the intended server is not the end point of the client-server tunnel that has been established. The proxy or firewall examines the data to ensure that it does not pose a security risk to the private network. However, where the data is encrypted, the proxy or firewall cannot examine the data before passing it to the server, which is a security risk. It is noted that a proxy refers to an application sitting on the junction of traffic between different networks. A proxy may have several functions, such as managing network traffic, providing security, and enhancing network performance by caching.

[0005] Furthermore, the exchange of data in a secure manner requires overhead that some clients may not have. Establishing an SSL session, for example, requires that clients have the necessary protocol stack built or programmed into them. The encryption of data requires a fair amount of processing power as well. Certain clients, however, may not have the protocol built or programmed in, and/or may not have the requisite processing power to establish an SSL session in a usable way. For example, cellular phones, personal digital assistants (PDA's), and other clients that are referred to as "thin" clients, may be limited in these ways. This means that these clients may not be able to communicate securely over the Internet. This is problematic, since such thin clients are thought to be a new way for consumers to participate in electronic commerce.

[0006] For these and other reasons, therefore, there is a need for the present invention.

Summary of Invention

[0007] The invention relates to bi-directional multiple hopping via a proxy. The invention has two scenarios, a reverse scenario and a forward scenario, which can also be combined. In the reverse scenario, there is a client, one or more proxies, and an origin, or publishing, server. The term origin, or publishing, server is used to distinguish this server from the proxies, which may also be servers. The client sends encrypted data to the proxy over an unsecure network, such as the Internet, in a first hop. The proxy decrypts the encrypted data, and performs an action, or test, relative to the data, such as ensuring that the data does not present a security risk, and offering the benefit of redirecting the traffic as appropriate. Assuming that the data passes this test, the proxy can do one of two things. First, the proxy can send the data over a secure network of which it is a part, such as an intranet or other private network, to an origin server that is also part of the secure network, in a second hop. Second, the proxy can first re-encrypt the data, and send the data either over the secure network or the unsecure network to a server or another proxy, also in a second hop.

[0008] The reverse scenario of the invention provides four advantages. The proxy is able to determine that the encrypted data sent by the client does not pose any security risks, such as including viruses, before forwarding the data on to the server. The reverse scenario simplifies traffic within the secure network, where the data is sent by the proxy to the server without additional encryption. This is because encryption causes performance degradation as a result of the additional overhead needed to encrypt the data. Where the data is sent within the secure network in a decrypted manner, this performance degradation is reduced. Data that is communicated between servers within the secure network and clients outside the secure network can also be cached by the proxy if so configured, which improves end-user performance. That is, the administrator may make a decision to improve performance by caching what was originally encrypted data, even though this may present a security risk.

[0009] Rather than having individual publishing servers within the secure network manage the keys required for clients to encrypt data intended for them, the proxy

can manage the keys for all the servers, simplifying public key management. Furthermore, the proxy can respond to multiple addresses, such as Internet Protocol (IP) addresses, where individual servers within the secure network may each have one or more such addresses. This latter advantages allows multiple SSL listeners to be configured on a single proxy. An SSL listener is an application that listens on ports that accept incoming connections on which data is encrypted using SSL.

[0010] In the forward scenario of the invention, there is also a client, a proxy, and an origin server. The client sends unencrypted data to the proxy over a secure network in a first hop. The client may be a thin client, such as a wireless phone or a personal digital assistant (PDA) device, and the secure network may be the wireless or carrier network for the wireless phone. The proxy in the carrier or wireless network performs an action, or test, relative to the unencrypted data, such as ensuring that the unencrypted data is acceptable for transmission. Assuming that the unencrypted data passes this test, the proxy encrypts the unencrypted data into encrypted data, and sends the encrypted data to the origin server over an unsecure network, such as the Internet, in a second hop.

[0011] The forward scenario also provides four advantages. Thin clients are able to send encrypted data over the Internet, even if they do not have this capability built into them. Even where the clients have this capability built in, the forward scenario simplifies traffic within the secure network, because the clients do not have to encrypt the data themselves. The proxy is able to determine that the data sent by the client is acceptable for transmission to the origin server. As in the reverse scenario, the proxy is also able to cache data exchanged between the client and the origin server.

[0012] The reverse scenario is reverse in that the proxy performs its decryption at the back end of the client-to-proxy-to-origin server communication path. Similarly, the forward scenario is forward in that the proxy performs its encryption at the front end of the client-to-proxy-to-origin server communication path. Whereas each scenario has been described as having two hops, they can also each have

more than two hops, where there is more than one proxy.

[0013] The invention is described in conjunction with a client-server scenario, in which there are clients, and servers, such as proxy servers and origin servers. However, the invention is applicable to other contexts as well, such as peer-to-peer environments. In the context of a peer-to-peer environment, for example, one of the peer nodes would at least temporarily function as one of the proxies as has been described, and other of the peer nodes would function at least temporarily as one of the servers or clients as have been described. That is, the description of the invention in relation to a client-server scenario is provided for descriptive and explanatory clarity only, and the invention is not to be construed to be limited to only this scenario. Where other contexts, such as a peer-to-peer scenario, lend their nodes to at least temporary functioning as the servers, proxies, and clients as described herein, these other contexts are also applicable to the invention. Furthermore, the hops as described in the invention are in the context of client-to-proxy, proxy-to-proxy, or proxy-to-server hops. That is, the hops are not hops between physical routers.

[0014] In addition to the embodiments, aspects, and advantages described in the summary, other embodiments, aspects, and advantages of the invention will become apparent by reading the detailed description and by referencing the drawings.

Brief Description of Drawings

[0015] FIG. 1 is a diagram showing an example two-hop reverse scenario according to the invention.

[0016] FIG. 2 is a flowchart of a method that one embodiment performs to implement the reverse scenario.

[0017] FIG. 3 is a diagram showing an example two-hop forward scenario according to the invention.

[0018] FIG. 4 is a flowchart of a method that one embodiment performs to implement

the forward scenario.

[0019] FIG. 5 is a diagram showing an example of a combined reverse and the forward scenario of the invention.

[0020] FIG. 6 is a diagram showing an example three-hop reverse scenario according to the invention.

[0021] FIG. 7 is a diagram of an example computerized device that can implement a client, a proxy, or an origin server as used in the invention.

Detailed Description

[0022] In the following detailed description of exemplary embodiments of the invention, reference is made to the accompanying drawings that form a part hereof, and in which is shown by way of illustration specific exemplary embodiments in which the invention may be practiced. These embodiments are described in sufficient detail to enable those skilled in the art to practice the invention. Other embodiments may be utilized, and logical, mechanical, electrical, and other changes may be made without departing from the spirit or scope of the present invention. The following detailed description is, therefore, not to be taken in a limiting sense, and the scope of the present invention is defined only by the appended claims.

[0023] Reverse Scenario

[0024]

The reverse scenario of the invention is depicted in the diagram 100 of FIG. 1. The client 104 is attempting to communicate securely with one of the origin servers 110, 112, and 114, over the Internet 102 via a publishing proxy 108. The origin servers 112 and 114 are part of a private network 106, of which the proxy 108 is also a part. The private network 106 can be, for example, an intranet. The private network 106 is a secure network. A secure network is one in which access is limited to a number of nodes, such that information can be exchanged among the nodes desirably without compromise. Because, for example, the client 104 is not part of the private network 106, this means that the client cannot access the

information that is exchanged among the proxy 108, the origin server 112, and the origin server 114. The proxy 108, the origin server 112, and the origin server 114 can communicate with one another in an unsecure manner, because the network 106 is itself secure.

[0025] By comparison, the Internet 102 is an example of an unsecure network. In an unsecure network, there is no guarantee that communications are not being intercepted or otherwise examined by others. For example, the client 104, when communicating with the proxy 108 over the Internet 102, cannot be sure that the communication is not being monitored by another node on the Internet 102. To ensure that that data being communicated between the client 104 and the proxy 108 is secure, it is encrypted. In this way, even if the data is monitored or intercepted by another node on the Internet 102, it cannot be compromised. That is, because the Internet 102 is itself unsecure, the client 104 communicates with the proxy 108 in a secure manner.

[0026] The client 104 establishes a secure connection between itself and the proxy 108 over the Internet 102, as represented by the arrow 116. The secure connection may be a secure socket layer (SSL) session, such that data is communicated between the client 104 and the proxy 108 according to the secure hypertext transport protocol (HTTPS). Other types of encryption may also be used. Because the connection between the client 104 and the proxy 108 is secure, this means that the data sent from the client 104 to the proxy 108 is also secure. In particular, the data is encrypted. The sending of encrypted data from the client 104 to the proxy 108 is performed in what is referred to as a first hop. The proxy 108 can alternatively be a firewall, or another type of device.

[0027] A hop as used herein is a data path from a beginning point to an ending point, where processing is performed on a data packet at the ending point, and not just routing and forwarding of the data packet. The processing can be encryption, decryption, examining the data, and so on. It is noted that this definition of hop is somewhat divergent from that used within the art, in where a hop is a data path between a beginning point to an ending pointing, where the ending point may

merely forward or otherwise route the data packet. That is, within a hop as this term is used in the patent application, there may be a number of hops as this term is used within the art. For example, in the sending of the encrypted data from the client 104 to the proxy 108, the encrypted data is likely to be routed and forwarded by a number of intermediary routers. These intermediary routers are not ending points of hops as the term is used herein.

[0028] The proxy 108 decrypts the encrypted data received from the client 104 into decrypted data. The proxy 108 then performs an action, or test, related to the decrypted data. For example, it may apply a security policy against the decrypted data, to determine if the data should be allowed to proceed to one of the origin servers 110, 112, and 114. Other actions are also possible. In general, the action performed by the proxy 108 against the decrypted data yields one of at least two results. First, the decrypted data may be deemed unacceptable, such that it is not allowed to pass through to one of the origin servers 110, 112, and 114. For example, the data may be deemed as presenting a security risk.

[0029] Another example includes the data having originated from a client 104 with which the intended one of the origin servers 110, 112, and 114 is not allowed to communicate. Alternatively, the data may be transmitted according to an acceptable communications protocol that is not allowed for communication by the intended one of the origin servers 110, 112, and 114. Other examples are also possible, and are included under the definition of the data being deemed unacceptable. In these cases, the proxy 108 may take other actions, such as returning an error message to the client 104 and/or the intended origin server, indicating that its data has been blocked. In addition, or alternatively, the proxy may take other actions, such as discarding the message without returning an error message, logging the incident as part of a security policy, sending notification to another system or individual about the incident, and so forth.

[0030]

Second, the decrypted data may be deemed acceptable, such that it is allowed to pass through to one of the origin servers 110, 112, and 114. For example, the data may be deemed as not presenting a security risk, the data may be deemed as

originating from an acceptable client 104, the data may be deemed as transmitted according to an acceptable communications protocol, and so on. Depending on the nature of the decrypted data, the proxy 108 then sends the data to one of the origin servers 110, 112, and 114, in what is referred to as a second hop. The sending of the data to the origin servers 110, 112, and 114 represents three different examples of how the data can be sent from the proxy 108 in the second hop. First, the proxy 108 can encrypt the data again and send the data to the origin server 110. Because the origin server 110 is not within the private network 106, the data must be again encrypted to ensure that it passes securely through the Internet 102. The proxy 108 can, for example, establish a new SSL connection with the origin server 110, and send the data over HTTPS to the origin server 110.

[0031] Second, the proxy 108 can send the decrypted data to the origin server 112, as represented by the arrow 120. Because the private network 106 is secure, the proxy 108 is able to send the data to the origin server 112 without encryption, and the data remains secure. For example, the data may be sent by the known hypertext transport protocol (HTTP). Third, the proxy 108 can encrypt the data again, and send the encrypted data to the origin server 114, as represented by the arrow 122. For example, the proxy 108 can establish an SSL connection with the origin server 114, and send the data over HTTPS to the origin server 114.

[0032] In summary, the diagram 100 of FIG. 1 illustrates the two-hop reverse scenario of the invention. In the first hop, the client 104 sends encrypted data to the proxy 108 over an unsecure network, the Internet 102. The proxy 108 decrypts the encrypted data into decrypted data, and performs an action, or test, relative to the data. If the data passes this test, the proxy 108 sends the data in the second hop. The proxy 108 can encrypt the data again and send it to the origin server 110 over the Internet 102, send the unencrypted data to the origin server 112 over the private network 106, or encrypt the data again and send it to the origin server 114 over the private network 106. The scenario of the diagram 100 is reverse in that the decryption by the proxy 108 occurs at the back end of the client-to-proxy-to-origin server communication path.

[0033] The two-hop reverse scenario ensures that the proxy 108 has an opportunity to examine the encrypted data sent by the client 104 before it is passed along to one of the origin servers 110, 112, and 114. This can ensure the overall security of the private network 106 and/or the origin servers 110, 112, and 114. The proxy 108 can send the decrypted data without further encryption in the second hop, such as to the origin server 112. In this case, the proxy 108 is able to manage the public encryption keys for origin servers such as the origin server 112, which renders key maintenance more easily managed. Furthermore, traffic within the network 106 is simplified in this case, because data does not have to be encrypted when it is sent within the network 106. Each of the origin servers 110, 112, and 114 may have a separate address for communication with by clients such as the client 104, such as a separate Internet Protocol (IP) address. In this case, the proxy 108 is able to handle multiple addresses so that it initially receives data intended for one of the origin servers 110, 112, and 114. The proxy 108 can also act as a cache for data communicated between the client 104 and the origin servers 110, 112, and 114, decreasing the burden placed on the origin servers.

[0034] A summary of the reverse scenario of the invention is also made with reference to the method 200 of the flowchart of FIG. 2. The method 200 is divided into columns 202, 204, and 206, indicating that parts of it are performed by the client 104, the proxy 108, and one of the origin servers 110, 112, and 114, respectively. The client 104 sends a secure request to the proxy 108 (208). The proxy 108 receives the request (210), and decrypts the request (212).

[0035] The proxy 108 next determines whether the request is acceptable for transmission to one of the origin servers 110, 112, and 114 (214). For example, the proxy 108 performs an action, or test, relative to the decrypted request to ensure that the request does not present a security risk. If the request is acceptable, the proxy 108 forwards the request to one of the origin servers 110, 112, and 114 (216). This may or may not include encrypting the request again. The intended origin server then receives the requested forwarded by the proxy 108 (218) , as indicated by the arrow 217, in an encrypted or unencrypted manner. If the request is unacceptable, the proxy 108 sends an error message to the client

104 (220), which receives the error message (222).

[0036] Acceptability or unacceptability of the request can be determined by any of a number of different tests besides whether or not the request poses a security risk. The originating client 104 may be on an acceptable or unacceptable list, such that the intended one of the origin servers 110, 112, and 114 is or is not allowed to communicate with such a client 104. The request may have been transmitted according to a communications protocol that is or is not acceptable. Other examples of testing are also included in determining of the acceptability or unacceptability of the request.

[0037] Forward Scenario

[0038] The forward scenario of the invention is depicted in the diagram 300 of FIG. 3. The client 306 is attempting to communicate securely with the origin server 310. The client 306 is part of a secure network 304, of which the proxy 308 is also a part. The secure network 304 can be a carrier or a wireless network for wireless phones, for example, or another type of network. In the case where the secure network 304 is a carrier or a wireless network, the client 306 may be a wireless phone, or a device with wireless communication capability, such as a personal digital assistant (PDA) device, laptop or notebook computer, or another type of device. The client 306 can be a thin client, which is a client that has limited processing and computing ability. For example, a thin client may not be able to accommodate the necessary processing overhead for encrypting data, and otherwise performing secure communications. As another example, the thin client may perform encryption too slowly, rendering session establishment time unacceptable for the end user. Furthermore, alternatively, the client 306 may reside in a separate network than the proxy 308. For example, the client 306 may reside in a corporate network that is connected to the secure network 304 in which the proxy 308 resides.

[0039] The client 306 sends unencrypted data to the proxy 308 over the secure network 304. The data may be sent according to the hypertext transport protocol (HTTP), the wireless access protocol (WAP), or another protocol, such as a mail

protocol like the post office protocol (POP), or the Internet messaging access protocol (IMAP). The sending of the unencrypted data from the client 306 to the proxy 308 is performed in what is referred to as a first hop. The first hop is represented in FIG. 3 by the arrow 312. The client 306 can communicate with the proxy 308 in an unsecure, unencrypted manner, because the network 304 is itself secure.

[0040] The proxy 308 receives the unencrypted data from the client 306. The proxy 308 then performs an action, or test, related to the unencrypted data. For example, it may apply a security policy against the decrypted data, to determine if the data should be allowed to proceed to the origin server 310. Other actions are also possible. In general, the action performed by the proxy 308 against the unencrypted data yields one of at least two results. First, the decrypted data may be deemed unacceptable, such that it is not allowed to pass through to the origin server 310. For example, the client 306 may not be allowed to access the origin server 310. In this case, the proxy 308 may take other actions, such as returning an error message to the client 306 and/or the intended origin server, indicating that its data has been blocked. In addition, or alternatively, the proxy may take other actions, such as discarding the message without returning an error message, logging the incident as part of a security policy, sending notification to another system or individual about the incident, and so forth.

[0041] Second, the unencrypted data may be deemed acceptable, such that it is allowed to pass to the origin server 310. For example, the client 306 may be allowed to access the origin server 310. In this case, the proxy 308 establishes a secure connection between itself and the origin server 310 over the Internet 102, as represented by the arrow 314. The secure connection may be a secure socket layer (SSL) session, such that data is communicated between the proxy 308 and the origin server 310 according to the secure hypertext transport protocol (HTTPS). Because the connection between the proxy 308 and the origin server 310 is secure, this means that the data sent from the proxy 308 to the proxy 308 is secure.

[0042]

In particular, the data is encrypted. The sending of the encrypted data from the

proxy 308 to the origin server 310 is performed in what is referred to as a second hop. It is noted that the proxy 308 may also communicate with the origin server 310 in an unencrypted, unsecure manner. For example, the proxy 308 may communicate with the origin server 310 according to HTTP, or another unsecure protocol.

[0043] In summary, the diagram 300 of FIG. 3 illustrates the two-hop forward scenario of the invention. In the first hop, the client 306 sends unencrypted data to the proxy 308 over the secure network 304, such as a carrier or a wireless network. The proxy 308 performs an action, or test, relative to the data. If the data passes this test, the proxy 308 sends the data in the second hop. The proxy 308 in particular encrypts the data, and sends it to the origin server 310 over the Internet 102. Alternatively, the proxy 308 may choose not to encrypt the data, and send the unencrypted data to the origin server 310 over the Internet 310. The scenario of the diagram 300 is forward in that the encryption by the proxy 308 occurs at the front end of the client-to-proxy-to-origin server communication path.

[0044] The two-hop forward scenario ensures that the proxy 308 has an opportunity to examine the unencrypted data sent by the client 306 before it is passed to the origin server 310. This enables the proxy 308 to enforce security, access, and other policies of the secure network 304. The two-hop scenario enables thin clients that do not have the ability to encrypt data to otherwise communicate in a secure manner with the origin server 310 over the Internet 102. Even where the clients have the ability to encrypt data, moving the performance of the encryption from the clients to the proxy 308 simplifies and reduces traffic within the secure network 304. The proxy 308 can also act as a cache for data communicated between the client 306 and the origin server 310, decreasing the burden placed on the origin server 310.

[0045]

A summary of the forward scenario of the invention is also made with reference to the method 400 of the flowchart of FIG. 4. The method 400 is divided into columns 402, 404, and 406, indicating that parts of it are performed by the client 306, the proxy 308, and the origin server 310, respectively. The client 306 sends

an unsecure request to the proxy 308 (408). The proxy 308 receives the request (410), and determines whether the request is acceptable for transmission to the origin server 310 (412). For example, the proxy 308 performs an action, or test, relative to the unencrypted request to ensure that the client 306 is permitted to access the origin server 310. If the request is acceptable, the proxy 308 encrypts the request, and sends the resulting secure request to the origin server 310 (414). The origin server 310 then receives and decrypts the secure request (416), as indicated by the arrow 415, which can be in an unencrypted or encrypted manner. If the request is unacceptable, the proxy 308 sends an error message to the client 306 (418), which receives the error message (420).

[0046] As has been indicated, acceptability or unacceptability of the request can be determined by any of a number of different tests besides whether or not the request poses a security risk. The originating client 306 may be on an acceptable or unacceptable list, such that the origin server 310 is or is not allowed to communicate with such a client 306. The request may have been transmitted according to a communications protocol that is or is not acceptable. Other examples of testing are also included in determining of the acceptability or unacceptability of the request.

[0047] Combined Reverse and Forward Scenario

[0048] The reverse and forward scenarios can be combined, as is shown in the diagram 500 of FIG. 5. This is also referred to as the "end to end" scenario. The forward scenario is indicated by the curly brace 502, whereas the reverse scenario is indicated by the curly brace 504. For the reverse scenario, the client 104 is an effective client, in that the client 306 and the proxy 308 that are a part of the secure network 304 are effectively the client 104 in the reverse scenario. Likewise, for the forward scenario, the origin server 310 is an effective origin server, in that the proxy 108 and the origin server 112 that are a part of the private network 106 are effectively the origin server 310 in the reverse scenario. That is, an effective client is a proxy and a client within a network that appears as a client, whereas an effective origin server is a proxy and an origin server within a network that appears

as an origin server.

[0049] The combined reverse and forward scenario operates as follows. In the forward scenario, the client 306 sends unencrypted data over the secure network 304 to the proxy 308, as indicated by the arrow 312. This is the first hop of the forward scenario. The proxy 308 performs an action, or test, relative to the data, and if the data passes, encrypts the data and sends it over the Internet 102 (or other unsecure network) to the proxy 108, as indicated by the arrow labeled 116 and 314. This is the second hop of the forward scenario. Note that the proxy 308 intends the encrypted data to be sent to the origin server 112, but the encrypted data is in fact sent to the proxy 308. Thus, it can be stated that the proxy 308 sends the encrypted data to the effective origin server 310.

[0050] The second hop of the forward scenario is also the first hop of the reverse scenario. In the reverse scenario, the proxy 108 receives the encrypted data from the proxy 308, as indicated by the arrow labeled 116 and 314. Note that the proxy 108 believes that it is received the encrypted data from the client 306, but the encrypted data is in fact received from the proxy 308. Thus, it can be stated that the proxy 108 receives the encrypted data from the effective client 104. The proxy 108 decrypts the encrypted data into decrypted data, and performs an action, or test, relative to the data. If the data passes, the proxy 108 sends the unencrypted data over the private network 106 to the origin server 112, as indicated by the arrow 106. This is the second hop of the second scenario.

[0051] It is noted that there is a total of three hops in the combined forward and reverse scenario of the diagram 500 of FIG. 5. The first hop of the forward scenario, as indicated by the arrow 312, is the first hop of the combined forward and reverse scenario. The second hop of the forward scenario, which is also the first hop of the reverse scenario, as indicated by the arrow labeled 116 and 314, is the second hop of the combined forward and reverse scenario. The second hop of the reverse scenario, as indicated by the arrow 120, is the third hop of the combined forward and reverse scenario.

[0052] Multiple Hops Within a Single Reverse or Forward Scenario

[0053] The reverse scenario and the forward scenario have been described as having two hops. The combined reverse and forward scenario has been described as having three hops. However, individual reverse or forward scenarios can also have more than two hops. Generally, the invention provides for a multiple-hop reverse, forward, or combined scenario. As an example of a specific multiple-hop reverse, forward, or combined scenario, a three-hop reverse scenario is depicted in the diagram 600 of FIG. 6.

[0054] In the first hop 610, the client 104 sends encrypted data to the proxy 108 over the Internet 102, as indicated by the arrow 116. The proxy 108 decrypts the encrypted data into decrypted data, and performs an action, or test, relative to the data. If the data passes, then the proxy 108 again encrypts the data. In the second hop 612, the proxy 108 sends the again encrypted data to the proxy 602 over the Internet 102, as indicated by the arrow 118. Note that the proxy 108 intends the again encrypted data to be sent to the origin server 608, but the encrypted data is in fact sent to the proxy 602. Thus, it can be stated that the proxy 108 sends the encrypted data to the effective origin server 110.

[0055] The proxy 602 receives the again encrypted data from the proxy 108. Note that the proxy 108 believes that it received the encrypted data from the client 104, but the encrypted data is in fact received from the proxy 108. Thus, it can be stated that the proxy 602 receives the again encrypted data from the effective client 616. The proxy 602 decrypts the again encrypted data into again decrypted data, and performs an action, or test, relative to the data. If the data passes, then the proxy 602 sends the again decrypted data to the origin server 608 over the private network 606, as indicated by the arrow 604. This is the third hop 614.

[0056] Example Computerized Device

[0057]

The invention can be implemented within a computerized environment having one or more computerized devices. The diagram of FIG. 7 shows an example computerized device 700, which can act as a client or a server in the invention. The example computerized device 700 can be, for example, a desktop computer, a

laptop computer, a cellular phone, or a personal digital assistant (PDA). The invention may be practiced with other computer system configurations as well, including multiprocessor systems, microprocessor-based or programmable consumer electronics, network computers, minicomputers, and mainframe computers. The invention may be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network.

[0058] The device 700 includes one or more of the following components: processor (s) 702, memory 704, storage 706, a communications component 708, input device(s) 710, a display 712, and output device(s) 714. For a particular instantiation of the device 700, one or more of these components may not be present. For example, a PDA may not have any output device(s) 714. The description of the device 700 is to be used as an overview of the types of components that typically reside within such a device, and is not meant as a limiting or exhaustive description.

[0059] The processor(s) 702 may include a single central-processing unit (CPU), or a plurality of processing units, commonly referred to as a parallel processing environment. The memory 704 may include read-only memory (ROM) and/or random-access memory (RAM). The storage 706 may be any type of storage, such as fixed-media storage devices and removable-media storage devices. Examples of the former include hard disk drives, and flash or other non-volatile memory. Examples of the latter include tape drives, optical drives like CD-ROM drives, and floppy disk drives. The storage devices and their associated computer-readable media provide non-volatile storage of computer-readable instructions, data structures, program modules, and other data. Any type of computer-readable media that can store data and that is accessible by a computer can be used.

[0060]

The device 700 may operate in a network environment. Examples of networks include the Internet, intranets, extranets, local-area networks (LAN's), and wide-area networks (WAN's). The device 700 may include a communications component 708, which can be present in or attached to the device 700. The component 708

may be one or more of a network card, an Ethernet card, an analog modem, a cable modem, a digital subscriber loop (DSL) modem, and an Integrated Services Digital Network (ISDN) adapter. The input device(s) 710 are the mechanisms by which a user provides input to the device 700. Such device(s) 710 can include keyboards, pointing devices, microphones, joysticks, game pads, and scanners. The display 712 is how the device 700 typically shows output to the user. The display 712 can include cathode-ray tube (CRT) display devices and flat-panel display (FPD) display devices. The device 700 may provide output to the user via other output device(s) 714. The output device(s) 714 can include speakers, printers, and other types of devices.

[0061] The methods that have been described can be computer-implemented on the device 700. A computer-implemented method is desirably realized at least in part as one or more programs running on a computer. The programs can be executed from a computer-readable medium such as a memory by a processor of a computer. The programs are desirably storable on a machine-readable medium, such as a floppy disk or a CD-ROM, for distribution and installation and execution on another computer. The program or programs can be a part of a computer system, a computer, or a computerized device.

[0062] Conclusion

[0063] It is noted that, although specific embodiments have been illustrated and described herein, it will be appreciated by those of ordinary skill in the art that any arrangement is calculated to achieve the same purpose may be substituted for the specific embodiments shown. This application is intended to cover any adaptations or variations of the present invention. Therefore, it is manifestly intended that this invention be limited only by the claims and equivalents thereof.